Спосіб Логічного Асоціативного Зв'язування Ресурсів у Неструктурованих Текстових Сховищах

Тарасенко В.П., Михайлюк А.Ю., Петрашенко А.В., Замятін Д.С.

Кафедра Спеціалізованих комп'ютерних систем, Національний технічний університет України "Київський політехнічний інститут", вул. Політехнічна, 14, корп. 15, Київ, 03056, УКРАЇНА,
E-mail: dsz@ukr.net

Abstract – In this article we developed the approach to logic association of data resources on unstructured warehouses which is based on document templates. The document template is a data structure for document properties definition which is independent of physical database. It is shown that the template can be presented in disjunctive normal form of logical predicates of the document attributes. We propose the algorithm to obtain the documents set which conforms with given template. We also obtained the analytical estimation of computational complexity for discussed algorithm implementation using SSE2 vector extensions and NVidia CUDA.

Key words – unstructured text warehouses, document templates, logic associations, GPGPU.

I. Вступ

Зростання об'ємів інформації, яка зберігається у інформаційних базах ізольованих автоматизованих систем, що застосовуються для підтримки бізнес-процесів підприємств та організації, з часом значно знижує ефективність комплексної обробки даних і, відповідно, негативно впливає на продуктивність праці персоналу. Тому задача поєднання окремих електронних ресурсів, дані в яких мають приховані зв'язки, у сховища та засоби обробки даних у подібних сховищах є надзвичайно актуальною.

В даній роботі пропонується спосіб визначення наборів документів, які обробляються за специфічними алгоритмами, лише на основі значень атрибутів, що дозволяє створювати системи, поведінка яких визначається вхідними даними.

II. Постановка задачі

Нехай, сховище W складається з множини документів D_i, $i = 1...n$, A — множина всіх можливих атрибутів документів $D_i \in W$, тоді документ D_i можна подати як кортеж, що складається з множини атрибутів $A_j \in A$ документа та їх значень V_j:

$$D_i = <A_j : V_j>$$

Таким чином, неструктуроване текстове сховище можна подати як множину документів, кожний з яких складається з множини значень певних атрибутів [1].

Досить типовою є ситуація, коли певна підмножина документів сховища W вимагає обробки за допомогою певних специфічних алгоритмів. Прикладами таких ситуацій можуть бути наступні задачі:

– формування списку атрибутів для документа, який буде створено;

– визначення порядку атрибутів та їх розташування на екрані при візуалізації документів, які вже існують;

– розрахунок значень агрегатних функцій за набором документів, які мають вказаний набір атрибутів (наявність документів без вказаних атрибутів буде призводити до помилкових значень розрахунків, наприклад, середнього значення);

– визначення набору документів, які може переглядати чи модифікувати певний користувач (задача розмежування доступу);

– тощо.

Вживаним архітектурним патерном, який використовується в подібних ситуаціях, є створення окремого атрибуту, який характеризує документ за його приналежністю до певної групи документів, тобто за його типом. Складністю застосування такого підходу є зниження гнучкості програмних засобів, що розробляються, через можливість появи додаткових підтипів у існуючих типів документів, а також появи нових типів. Для кожного з таких підтипів потрібно реалізовувати специфічні алгоритми обробки, при чому внесення нового програмного коду буде відбуватися одразу в багатьох місцях програми, зокрема при візуалізації, пошуці, каталогізації даних, а також у всіх специфічних режимах. В результаті це призводитиме до ускладнення архітектури системи та збільшення необхідних зусиль для тестування та подальшої розробки.

Таким чином, умовою розробки експлуатаційно придатної системи підтримки сховища документів є наявність механізмів групування документів за типами.

III. Шаблони документів

Очевидно, значення суттєвих атрибутів документу залежать від його класу у предметній області, і, тому, здатні визначити його тип. Якщо набору існуючих атрибутів не достатньо для того, щоб відрізнити один тип документу від іншого, можна ввести додатковий атрибут, який відображатиме те, що в предметній області розглянуті документи відрізняються за типами. Будемо вважати, що тип документа визначається значеннями підмножини атрибутів $\{A'_1, A'_2, ..., A'_n\} \in A$. Нехай V^A — це значення атрибуту A, V — певна константа, а характеристична булева функція $f(V^A, V)$ повертає істинне значення, якщо

V^A та V знаходяться у відповідному відношенні. Наприклад, якщо значення атрибуту A відносяться до числового типу, функція f може бути задана як

$$f(V^A, V) = V^A \theta V, \qquad (1)$$

де $\theta \in \{=, \neq, >, <, \leq, \geq\}$. Оскільки тип документа визначається значеннями його атрибутів, то приналежність документа до заданого типу визначатиметься булевою функцією $F(f_1, f_2, ..., f_n)$, де $f_1, f_2, ..., f_n$ — відповідні характеристичні функції для атрибутів $A'_1, A'_2, ..., A'_n$, яку ми назвемо *шаблоном*.

Відомо, що будь-яку булеву функцію можна подати у вигляді диз'юнктивної нормальної форми (ДНФ), тобто у вигляді диз'юнкції кон'юнкцій вхідних параметрів, кожна з яких визначає певну конституенту одиниці. Тобто, шаблон типу документа F можна задати як

$$F(f_1, f_2, ..., f_n) = \bigvee_{i=1..k}(\bigwedge_{j=1..n} f_j), \qquad (2)$$

де k — кількість конституент одиниці.

Виходячи з визначення сховища документів та поширивши для сховищ відомі теоретико-множинні операції можна довести, що диз'юнкція характеристичних функцій відповідає об'єднанню сховищ. Аналогічно теж саме можна довести і для кон'юнкції та перетину. Звідси можна запропонувати алгоритм отримання набору документів за певним шаблоном (рис. 1).

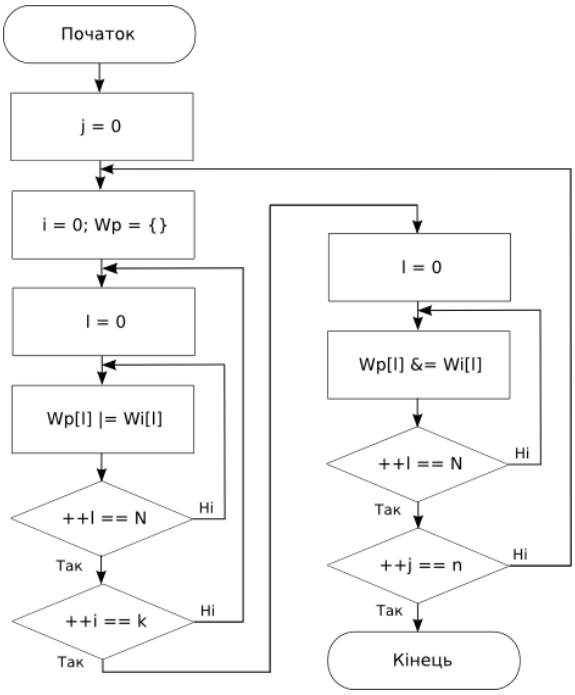

Рис.1 Схема алгоритму отримання набору документів, що відповідають заданому шаблону.

IV.Шаблони у реляційному поданні

На сьогодні фактичним стандартом розробки програмних засобів обробки документів у сховищах стало застосування трирівневої архітектури, в якій в основі шару збереження інформації використовується реляційна система управління базами даних [2]. Тому для уніфікації інтерфейсів доступу до структур даних, які використовуються програмними засобами, доцільно розробити ефективне подання шаблонів документів у вигляді таблиці реляційної бази даних (БД).Визначення характеристичної функції, надане у (1) можна також поширити на рядкові дані та дані типу дата шляхом введення для них функцій порівняння на основі впорядкованості у лексографічному або у хронологічному порядку відповідно.

Отже, характеристична функція може бути подана як кортеж (f, A, θ, V), де f — ідентифікатор характеристичної функцій, A - ідентифікатор атрибуту, θ - оператор з множини $\{=, \neq, >, <, \leq, \geq\}$,$V$ - константа, з якою відбувається порівняння.

На рис. 2 наведено структуру сегменту БД, який відповідає запропонованому поданню шаблонів, у вигляді діаграми класів UML. Фрагмент БД, який призначено для збереження атрибутів документів та їх значень побудовано на основі моделі Entity-Attribute-Value, що надає можливість зберігати документи довільної структури.

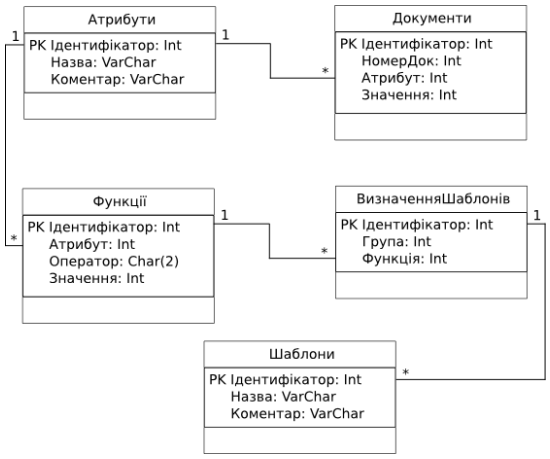

Рис.2 UML-діаграма структури сегмента БД, який відповідає за збереження шаблонів.

V.Параметри швидкодії механізму шаблонів

Застосування механізму шаблонів вимагає додаткових витрат пам'яті та обчислювальних ресурсів, що може призвести до зниження часу відклику систем, що надають результати обчислень в реальному часі.

Основу механізму шаблонів складають характеристичні функції, реалізація яких полягає у застосуванні функції фільтрації до всього масиву документів у сховищі. Фільтрація документів за значенням певного атрибуту вимагає проходу по всім документам, тобто, при кількості документів у сховищі N, складність одноразового застосування функції складатиме $\Theta(N)$. Якщо у визначенні шаблону використано M характеристичних функцій, складність зростає до $\Theta(M \times N)$.

Збільшити швидкодію операції визначення значень характеристичних функцій можна за рахунок їх попереднього обчислення. Оскільки кожна функція повертає значення з множини {True, False}, результат її застосування до всього

масиву документів можна подати як бітовий рядок, де кожний біт відповідає певному документу, тобто використати так звані бітові індекси [3].

Оскільки, при застосуванні бітових індексів, за значення кожної окремої характеристичної функції відповідає 1 біт, швидкодія розрахунку буде залежати від розрядності машинного слова. При кількості документів у сховищі N, якщо кількість диз'юнкцій у шаблоні становить K_D, а кон'юнкцій K_K, кількість кроків, необхідних для розрахунку бітового подання одного масиву складатиме:

$$\sum_{i=1}^{K_D+K_K} \frac{N}{R} = \frac{1}{R} \sum_{i=1}^{K_D+K_K} N,$$

де R — кількість розрядів машинного слова. Зокрема, при застосуванні процесорів з 64-бітною архітектурою, кількість операцій, потрібних для розрахунку буде $N/64$, а при використанні векторних розширень SSE2, які дозволяють виконувати бітові операції над 128-бітними числами — $N/128$.

Останнім часом, значне поширення отримали процесори з багатоядерною архітектурою, використання яких дозволяє зменшити кількість потрібних для розрахунку кроків ще у P разів, де P — кількість ядер у процесорі:

$$\frac{1}{RP} \sum_{i=1}^{K_D+K_K} N$$

При значних обсягах обчислень за шаблонами, для зменшення навантаження на центральний процесор, можуть також бути застосовані гетерогенні системи, наприклад з використанням технологій GPGPU на основі NVidia CUDA або OpenCL. В такому випадку, крім часу, який витрачатиметься безпосередньо на розрахунок, потрібно також врахувати час на перенесення результату з пам'яті графічного пристрою, який в загальному випадку, є пропорційним N. Час на завантаження бітових індексів в пам'ять пристрою можна не враховувати, оскільки ця операція виконується одноразово перед початком обчислень. У випадку застосування технологій GPGPU, кількість операцій для розрахунку складатиме:

$$\frac{1}{RP} \sum_{i=1}^{K_D+K_K} N + \Theta(N)$$

В цьому випадку слід враховувати, що значення P значно більше, ніж при застосуванні багатоядерних процесорів.

VI.Застосування механізму шаблонів

Механізм шаблонів, розглянутий у даній роботі було впроваджено при розробці інформаційно-аналітичної системи, призначеної для аналізу та систематизації сховища текстовміщуючих ресурсів Київського університету ім. Б. Грінченка. Система розроблялася як web-орієнтована з використання технології AJAX на основі бібліотеки Jquery. Для серверної підсистеми було використано мову програмування Python та фреймворк Django. Для створення та редагування шаблонів було розроблено спеціалізовані візуальні засоби, які за допомогою контекстних підказок дозволяють зменшити ймовірність технічних помилок при введені значень атрибутів та гарантують цілісність даних у шаблонах.

Впровадження механізму шаблонів показало його доцільність, перш за все за рахунок можливості ефективної реалізації у вигляді надбудови над ядром системи, тому в подальшому планується інтегрувати цей механізм з іншими функціональними режимами системи.

Висновки

В роботі розглянуто задачу групування документів у сховищах для подальшого застосування спільних алгоритмів обробки та запропоновано спосіб її вирішення, оснований на механізмі шаблонів. Механізм шаблонів документів базується на використанні значень атрибутів, що дозволяє при його реалізації не вводити додаткових спеціалізованих атрибутів, тобто не змінювати модель подання документів в системі. Шаблон визначається як булева функція, яка в якості аргументів отримує значення характеристичних функцій для окремих документів та подається у вигляді диз'юнктивної нормальної форми.

Для реалізації механізму шаблонів у вигляді програмної компоненти, запропоновано спосіб подання шаблонів у реляційних базах даних та способи підвищення швидкодії алгоритму розрахунку сукупності документів на основі бітових індексів та розпаралелення з використанням багатоядерних процесорів та технологій GPGPU. На прикладі впровадження механізму шаблонів у систему обробки корпоративного сховища документів показано його ефективність.

Література

[1] Mykhailyuk A., Zamiatin D., Petrashenko A. Unstructured Data Warehouse Processing System Based on an Uniform Set of Functions // Proceedings of the 4-th International Conference ACSN-2009 "Advanced Computer Systems and Networks: Design and Application". — Lviv. — 2009. — P. 117-119.

[2] Kadav A., Kawale J., Mitra P. Data Mining Standards, 2008, 33 p. [Електронний ресурс]. – Режим доступу: http://www.datamininggrid.org/wdat/works/att/standard01.content.08439.pdf

[3] O'Neil, P., Quass, D. *Improved Query Performance with Variant Indexes // ACM International Conference on Management of Data (SIGMOD 1997). — Tucson, Arizona. — 1*